208. Responsive to receiving a request for an action, Decision Engine **202** consults the current model and provides a recommended action to Decision Agent **108**. Decision Agent **108** performs the recommended Action. In response to the performed action in the environment, Decision Agent **108** provides a response/reward to continual learning system **102**.

[0024] In some exemplary embodiments, Decision Engine **202** notifies Archiver **204** that an action has been sent to Decision Agent **108** and Decision Agent **108** sent a response to the action. Archiver **204** writes transaction information to Transaction Database **206**. In other exemplary embodiments, Archiver **204** can buffer transactions until a time or size limit is reached and then write to Transaction Database **206** in a bulk fashion.

[0025] In some exemplary embodiments, Orchestrator **212** is the central coordinator of continual learning system **102**. Orchestrator **212** may contain one or more parameters, in which the parameters consist of an observation period and a test period. Orchestrator **212** may initiate the observation period and the test period. In an exemplary embodiment, the observation period may be the amount of time Decision Engine **202** uses a first model to prescribe actions for requests from Decision Agent **108**. Archiver **204** writes transaction data to Transaction Database **206** during the observation period. In an exemplary embodiment, the test period is the amount of time Decision Engine **202** uses the first model and a second model to prescribe actions for requests from Decision Agent **108**. In other exemplary embodiments, Decision Engine **202** can use more than two models during the test period. In yet other exemplary embodiments, Orchestrator **212** can be configured with a parameter to indicate the relative split of traffic across the models used during the test period, a parameter to specify the confidence level when comparing the performance of different models, and a parameter corresponding to the desired threshold to compare the difference in the average performance of models.

[0026] In some exemplary embodiments, Orchestrator **212** initiates Model Builder **210** to build models. Model Builder **210** may be executed at any time. In an exemplary embodiment, Model Builder **210** reads transaction data from Transaction Database **206** and trains a new model through various machine learning techniques for reinforcement learning known in the art, such as. Q-Learning, SARSA, and SARSA (λ). Model Builder **104** stores the new model in Model Database **208** after one is created.

[0027] In some exemplary embodiments, evaluator **214** can be at least one of an A/B Testing Evaluator, a Bandit Evaluator, or a Simulation Evaluator. In an exemplary embodiment in which Evaluator **214** implements an A/B Testing Evaluator, Evaluator **214** reads transaction data from Transaction Database **206** and compares the performance of the models in use during the test period using statistical hypothesis testing methods. In another exemplary embodiment in which Evaluator **214** implements a Bandit Evaluator, Evaluator **214** uses a two-arm or multi-arm Bandit approach to prescribe changes to the relative weights associated with each of the models being tested during the test period. In yet another exemplary embodiment in which Evaluator **214** implements a Simulation Evaluator, Evaluator **214** reads transaction data from Transaction Database **206**, builds a model of the environment, and compares the performance of alternative models by testing them in the simulated environment.

[0028] FIG. **3** is a flowchart illustrating operational steps of continual learning system **102**, generally designated **300**, according to an exemplary embodiment. In some exemplary embodiments discussed herein, decision agent **108** can initiate continual learning system **102** by connecting to server **104** via network **106**. Continual learning system **102** implements the operational steps utilizing the components of continual learning system **102**, referring to FIG. **2** discussed above.

[0029] Orchestrator **212** deploys a first model (**302**). In some exemplary embodiments, Orchestrator **212** retrieves a first model from Model Database **208** and deploys the first model, for example an initial seed model, to Decision Engine **202**. Model Builder **210** may build the first model offline using transaction data collected from a legacy system, transaction data generated from a rule-based system designed by human experts, or any other method known in the art that uses available information. Model Builder **210** may store the first model in Model Database **208**. In some exemplary embodiments, the first model may be referred to as model A.

[0030] Orchestrator **212** initiates an observation period (**304**). In some exemplary embodiments, Orchestrator **212** initiates the observation period using an observation period parameter. The observation period may be configured for a predetermined time period (e.g. an hour, a day, two weeks, a year). During the observation period, decision engine **202** services requests from Decision Agent **108** using the first model. Archiver **204** writes transaction data, regarding the requested services in the observation period (i.e. requested actions, actions, responses/rewards to the provided actions), to Transaction Database **206**.

[0031] In some exemplary embodiments, during the observation period, continual learning system **102** periodically "wakes up" to determine if the observation period has expired. If continual learning system **102** determines the observation period has not expired, continual learning system goes back to "sleep" until the next predetermined polling time interval to determine if the observation period has expired. If continual learning system **102** determines the observation period has expired, continual learning system **102** proceeds as described below.

[0032] Responsive to the observation period ending, Model Builder **210** builds a second model (**306**). In some exemplary embodiments, Model Builder **210** builds a second model using transaction data collected during the observation period and stored in Transaction Database **206**. In an exemplary embodiment, the second model may be referred to as model B. In other exemplary embodiments, Model Builder **210** builds more than one new model.

[0033] Orchestrator **212** deploys the second model (**308**). In some exemplary embodiments, Orchestrator **212** retrieves the second model from Model Database **208** and deploys the second model to Decision Engine **202**. In some exemplary embodiments, Orchestrator **212** deploys the second model to a percentage of users of Decision Agent **108**. For example, the first model can be configured to 80% of the users and the second model can be applied to the remaining 20% of the users. In another exemplary embodiment, Orchestrator **212** deploys the second model to all users of Decision Agent **108**.